# Real-time Modeling Experiment

For this exercise, you will set up and run the WRF EMS for the purpose of making a real-time forecast. You will be using the wrf_autorun.pl routine initiated from a cron that will (hopefully) execute each of the steps from downloading the initialization files, processing the data, running the model, and post processing the forecasts. You will have nearly complete control over the configuration of your computational domain and configuration of the model run. Your only objective is to have a 24 hour forecast competed and available within four hours from the start of the process. Each group will initiate their runs overnight and do the analysis tomorrow. Each group will then present their forecasts to the class in a brief 10-minute presentation.

## Day 1: Set up and test the model run

***Important! - Testing to make sure your real-time run will be successful is critical for success of this exercise. Therefore you will be required make two short test forecasts before committing your forecast to the crontab file.***

### *Step I Create a computational domain*

Just like in the previous exercises, you need to create your computational domain using the static initialization GUI (SIGUI). Unlike the other exercises however, you will have complete control over the configuration of your computational domain. That said, it is important that you not get too ambitious. Remember that the goal it to get a 24 hour forecast done in less than four hours, *total*! *The four hour period includes the time required to download and process the initialization files, run the model, and then post process all the forecast files. Everything in four hours!*

**Question #1: How much time do you have from start to finish of your real-time simulation?**

To help you meet this quasi-arbitrary four hour time limit, here a few suggestions to consider when setting you your domain:

1. You will be using the 1 degree GFS "personal tiles" data set for model run initialization. The size of these files is smaller than the "gfsgrb2" data set you have been using for the previous experiments, yet retain the full spatial resolution. While the files reside on a remote server in the US, the bandwidth required to download this data set should not be an issue unless all everyone is trying to access the server at once.
2. Unlike the "gfsgrb2" data set, which is in GRIB 2 format, the personal tiles are in GRIB 1 format and take less time to process on your system.
3. You may choose to run the ARW or NMM core for this experiment. Recall which core was faster in the benchmark case? This may factor into your decision.
4. Recall that a 2-fold increase in vertical levels requires a 2-fold increase in computation time, while a 2-fold increase in horizontal resolution (2-fold decreases in grid spacing) requires an 8-fold increase in computation time. You generally add more value to a forecast by increasing the horizontal resolution however. – It's your quandary.
5. Do not set up a nested domain for this experiment. Start with a simple configuration to improve your chances of success.
6. Do not compromise areal coverage for resolution! The area coverage should be the first thing you consider when setting up your computational domain. Afterwards, determine the grid spacing that you can afford to use over your forecast period (24 hours).
7. If you run the model with a grid spacing less than 5km you should use the ARW core. But chances are that you will not approach that resolution for this exercise.
8. Your choice of model physics will not have a large impact on time required to run the model although you should still select schemes that are appropriate for the resolution.

9. Hourly output of forecast files is better than 3-hourly output, but it takes 3 time longer to post process.

**Do: % sigui**

The name that you provide for your domain in the SIGUI can be whatever you like provided you can remember it.

When your domain is configured, go ahead and run the localization, which should take a few minutes. The time required to localize a model domain is proportional to the size of the domain. When the localization has completed, exit out of the SIGUI with a "yes" and "yes".

**Do: % cd $WRF_RUN/<your domain>**

## *Step II Test Download and Processing of the initialization files*

As stated earlier, you will be using the personal tiles (--dset gfsptiles) for the real-time forecast. Consequently, you will have to access the US National Weather Service (NWS), Science and Operational Officer (SOO), Science and Training Resources Center (STRC) data server via http. The SOO/STRC server will be available to you after the workshop until a new dedicated server becomes available within a few months.

You should test the amount of time required to download and process the initialization data set.

**Do: Run wrf_prep to test the download and processing of the grib files**

**Do: % wrf_prep --dset gfsptiles --http strc --sfcdset sstptile --shttp strc --length 24**

**Question #2: What is the meaning of each flag used in the wrf_prep command above? Hint: Look in the appropriate gribinfo.conf file for clues.**

| Option | Description |
|---|---|
| **--dset gfsptile** | |
| **--http strc** | |
| **--sfcdset sstptile** | |
| **--shttp strc** | |

**Question #3: How does wrf_prep interpret the exclusion of the "--date" and the "--cycle" options in the command line? Hint: Look in the appropriate gribinfo.conf file for clues. Hint #2: You should never use --date and --cycle for real-time modeling.**

Note the amount of time required to download and process the files. You can expect that it will take approximately the same time during your real-time run.

**Question #2: How much time did it take to complete wrf_prep?**

After wrf_prep has successfully completed you can move on to configuring and testing the wrf_run routine.

### *Step III Configure the Simulation*

While you may configure the model for your real-time forecast any way you desire, just leave the default values if you have any reservations about a setting. If, during the testing,

you have a problem then you should consider reverting back to the default values. *Remember, the default values are you friends.*

**Do: Determine if any changes are needed to the following files:**

**conf/wrf_run/run_physics.conf**

**conf/wrf_run/run_DMrun.conf**

**conf/wrf_run/run_io.conf**

**Question #3: Identify the following settings for your real-time run.**

| **Model Dynamics** | **Scheme** |
|---|---|
| Model Core (NMM or ARW) | |
| | |
| **Model Physics** | **Scheme** |
| Cumulus scheme | |
| Microphysics scheme | |
| PBL Scheme | |
| Land Surface Scheme | |
| | |
| **Model Output Information** | **Value** |
| Forecast Output Frequency | |
| Precip Accumulation Frequency | |

### *Step IV Make a short test run of your real-time forecast*

You will now make a brief test run of your real-time forecast over a 3-hour period. You will be able to get an approximation for the total amount of time for the 24 hour forecast by simply multiplying the time by eight. Note that your actual real-time forecast will require less time to complete as there is a small amount of overhead at the beginning of the forecast.

**Do: % wrf_run --SMDM --length 3h**

**Note the use of the "h" in "--length 3h"**

**Question #4: How many minutes did it take to complete wrf_run?**

**Amount of minutes x 8 =**

When wrf_prep has completed you can move on to configuring and testing wrf_post.

### *Step V Test the post processing of the forecast files*

Throughout the previous four exercises we have neglected a discussion of the wrf_post.pl configuration files. The wrf_post.pl routine along with the ancillary Fortran and C programs rely on various configuration files and tables to complete the task of processing your WRF forecast data. These files provide the default wrf_post.pl options and may be edited to meet the needs of the user. Additionally, some of these options can be overridden by command line arguments to wrf_post.

### *The wrf_post.conf file*

In the absence of any command-line options, wrf_post will use the conf/wrf_post/wrf_post.conf file in determining how to process your WRF forecast files. Current options include processing forecast files into GRIB 1 & 2, GEMPAK, GrADS, BUFR, and BUKIT formats. There are also options for sending files to other systems or archive. Again, most of the parameters contained in the wrf_post.conf file may be overridden by command line options. Please see "wrf_post --help" or "wrf_post --guide" for more information on the command line options.

### *The other configuration files*

In addition to the wrf_post.conf file, there are other configuration files in the conf/wrf_post directory used to further your post-processing experience. These files should be reviewed and edited as necessary. The files are:

| | |
|---|---|
| **post_archive.conf** | - Forecast Archive Options |
| **post_bufr.conf** | - Converting forecast files to BUFR format |
| **post_grads.conf** | - Options for processing GrADS files |
| **post_gempak.conf** | - Options for processing GEMPAK files |
| **post_grib.conf** | - Options for processing GRIB 1 and 2 files |

For this exercise you will only need to edit the wrf_post.conf file for the purpose of turning ON the generation of GrADS files.

**Do: Edit wrf_post.conf and set "GRADS = Yes"**

**Do: Review the post_grads.conf files but don't change anything**

**Do:** **Review the wrf_post options:**

**% wrf_post --help**

**Do:** **Review the wrf_post user guide:**

**% wrf_post --guide**

**Do:** **% wrf_post**

Notes:

a. The wrf_post routine will read the wrf_post.conf file and see the user's request for GrADS file processing.

b. It will read the post_grads.conf file for any additional directions.

c. GRIB file processing will automatically be turned on since GRIB files are needed for the creation of GrADS files.

**Question #5:** **How many minutes did it take to complete wrf_post?**

**Amount of minutes x 8 =**

**Question #6:** **Time to run wrf_prep** **_____ minutes**

**Time to run wrf_run x 8** **_____ minutes**

**Time to run wrf_post x 8** **_____ minutes**

**Total time for simulation (Approx)** **_____ minutes**

## *Step VI Test run of the wrf_autorun.pl routine*

The wrf_autorun.pl routine is used to execute the various WRF EMS run-time scripts in a controlled process. It is designed to run wrf_prep.pl, wrf_run.pl, and wrf_post.pl in succession for the purpose of making real-time WRF forecasts; however, it may additionally be used to run case studies. For real-time forecasting, there are various options to improve the reliability of your forecasts, all of which are described in the conf/wrf_autorun.conf file ad nauseum. There is an option to post-process the WRF forecast files concurrent with model execution, which is intended to get forecasts out to forecasters but it requires the use of a second workstation to distribute the workload. *You will not use the wrf_autopost.pl option for this exercise.*

### The wrf_autorun configuration file

The configuration file for wrf_autorun is located in the conf/wrf_autorun directory, which is where you would make any changes. For this exercise, the wrf_autorun.conf file has been pre-configured with the proper settings for your real-time run. Nonetheless, you should review the file and answer the questions provided below:

### Running wrf_autorun.pl from the command line

The master wrf_autorun.pl file resides in the wrf/strc ($WRF_STRC) directory. Just like the other run-time scripts, users will not run the wrf_autorun.pl script directly from this directory, but rather, will execute “wrf_autorun” from one of the domain directories.

**Question #7: Identify the following settings for your real-time run from the wrf_autorun.conf file.**

| Parameter | Scheme |
|---|---|
| Data set | |
| Land surface model data set | |
| Static surface data set | |
| Forecast hour to use for initial condition | |
| Boundary Condition Frequency | |
| Forecast Length | |
| Method of data acquisition and server | |
| Method of static data acquisition and server | |

As with the other WRF EMS run-time scripts, wrf_autorun includes a help menu and users guide. The help menu provides a brief summary of all the options available to wrf_autorun. To view the brief help menu:

**Do: % wrf_autorun --help**

And for the verbose users guide:

**Do: % wrf_autorun --guide**

At this point almost everything is configured to run your real-time simulation overnight, but before the job is placed in a cron you should run one more test using the wrf_autorun script.

**Do: % wrf_autorun --length 3**

When you reviewed the help menu ("--help") you should have noticed the –length flag, which is used to override the default value for the length of the run (24 hours). Nearly all the command line flags in wrf_autorun are used to override the default values defined in the wrf_autorun.conf file. For this test of the autorun routine you are reducing the length of the run to 3 hours to expedite the process. If it runs for three hours it should run for 24 hours.

**Question #8: How many minutes did it take to complete wrf_autorun?**

**Amount of minutes x 8 =**

If the total amount of time required to run wrf_autorun is well in excess of four hours, you will have to modify your configuration to reduce the time.

## *Step VII Running wrf_autorun.pl from cron*

The final step in preparing your real-time run for "prime time" is to create an crontab entry that will be initiated at the specified time. Fortunately, much of the work has already been completed for you as part of the WRF EMS installation process. First, a tcsh wrapper file, wrf_autorun-wrapper.csh, is provided in the wrf/strc ($WRF_STRC) directory to facilitate the running of wrf_autorun from a cron. In addition, a deactivated entry was placed in your

crontab file during installation. Using the wrf_autorun-wrapper.csh file will make sure that your environment variables are set correctly prior to running wrf_autorun.

If you review the entries in your crontab file:

**Do: % crontab –l**

You should see three entries related to the WRF EMS:

1. An entry for the wrf_updat.pl,
2. An Entry for the wrf_autonest-wrapper.csh, and
3. An entry for wrf_autorun-wrapper.csh that looks something like:

#16 10 * * * /usr1/wrf/strc/wrf_autorun-wrapper.csh --verbose --rundir /usr1/wrf/runs/**<your domain>** >& wrf_autorun.log 2>&1

Each of the entries will be deactivated meaning that there is a "#" symbol present at the beginning of the line.

The wrf_autorun-wrapper.csh file is simply a Tshell wrapper that is used to initialize the WRF EMS environment variables before running wrf_autorun.pl. Not executing wrf_autorun this way via cron will result in the model failing to run.

In order to configure the wrf_autorun crontab entry for real-time modeling:

1. Remove the leading "#"
2. Replace the "**<your domain>**" with the name you gave to your real-time domain, i.e., the directory under wrf/runs.
3. Set the time that you wish the run to **start**.** *This requires some knowledge of how to edit the time information in a crontab file. If the machine is using UTC*

*then also specify UTC in the crontab file. If the machine is using local time then do likewise in the crontab file. Also, the initial column is reserved for the start minute; the second entry is start hour. In the above example the start time is 1016 UTC or local time.*

You will be making one more test before leaving for the evening, just to make sure everything is working as expected.

**Do: Deactivate and configure the crontab entry and set the start time for 5 minutes from the current time**

**Also Do: Add "--length 3" to the entry for this test, i.e.,**

16 10 * * * /usr1/wrf/strc/wrf_autorun-wrapper.csh **--length 3** --verbose ...

**You MUST remove the "--length 3" entry before leaving for the evening!!**

After the time that the simulation should have started check the progress by tailing the wrf/logs/wrf_autorun.log. If the file does not exist then the cron failed to activate. If there were other problems they should be specified in the file.

****A comment on simulation start times**

You want to start a real-time simulation as soon as the initialization files are available on the server. Most of the initialization files are identified by the cycle time of the originating model from which they are derived. For example, the GFS runs four times per day with cycle times of 00, 06, 12, and 18 UTC. The forecast files for these runs are available some time after the cycle time, depending upon the time required to run the model and post process the forecast files into GRIB format. For the case of the GFS this "delay" is typically about four hours after a cycle time. Thus, the 12 UTC forecast files for the GFS will not be available until approximately 1600 UTC. This processing delay is build into the wrf_prep routine through the DELAY parameter in the _gribinfo.conf

files. The wrf_prep routine knows when files should be available on the server and you should take this fact into account when specifying the time to start your model run. If you plan on using the 1800 UTC cycle of the GFS to start your run then set the start time in the crontab files sometime after 2200 UTC. Again, this time should be specified in local time if your system uses local time.

**Do: Remove the "--length 3" entry from the crontab file and specify the start time for your forecast.**

You are done for the evening